Nonconvex Splitting for Regularized Low-Rank + Sparse Decomposition

Rick Chartrand, *Senior Member, IEEE*

***Abstract*—We develop new nonconvex approaches for matrix optimization problems involving sparsity. The heart of the methods is a new, nonconvex penalty function that is designed for efficient minimization by means of a generalized shrinkage operation. We apply this approach to the decomposition of video into low rank and sparse components, which is able to separate moving objects from the stationary background better than in the convex case. In the case of noisy data, we add a nonconvex regularization, and apply a splitting approach to decompose the optimization problem into simple, parallelizable components. The nonconvex regularization ameliorates contrast loss, thereby allowing stronger denoising without losing more signal to the residual.**

***Index Terms*—Algorithms, compressed sensing, optimization, principal component analysis, video signal processing.**

I. Introduction

THE field of compressive sensing has seen a proliferation of research developing algorithms that exploit *sparsity* to reconstruct images or signals from very few measurements [1]. Most such algorithms solve optimization problems containing nonsmooth objective functions, such as the ℓ^1 norm. Because nonsmooth optimization is typically computationally challenging, from the beginning there has been much emphasis on the development of efficient algorithms.

The desire for efficiency has motivated the use of *convex* optimization in particular [2], [3]. This eliminates the concern of converging to an undesirable local minimum. However, our previous work provides substantial evidence [4]–[7] that nonconvex optimization dramatically outperforms convex methods in theory and in practice, and can be implemented very efficiently [8].

More recently, the technology for sparsity-based optimization developed for compressive sensing has been applied to matrix optimization problems. A notable example is *matrix completion*, which attempts to reconstruct a matrix with only a small fraction of its entries known [9], [10]. The assumption that the matrix has a low rank plays the analogous role of sparsity in compressive sensing.

A further extension seeks to decompose a matrix D of high-dimensional data into a sum of two components, one having low rank, the other being sparse. This can be expressed as the following optimization problem:

$$\min_{L,S} \operatorname{rank}(L) + \lambda \|S\|_0, \text{ subject to } L + S = D. \quad (1)$$

Here $\|\cdot\|_0$ counts the number of nonzero entries, and $\lambda > 0$ is a tuning parameter. We can regard L as a low-dimensional description of the data, while S consists of discrepancies from that model, which can be interesting in their own right. Applications considered thus far include automated background removal in video [11], text analysis [12], and image alignment [13].

We can compare (1) to principal component analysis (PCA), which would compute the matrix L of desired rank that minimizes $\|D - L\|_F$, the Frobenius (or entry-wise Euclidean) norm of the residual. Because the second term of (1) penalizes only the number of discrepancies and not their size, the low-dimensional model L will not be perturbed by outliers among the entries of D, and hence will provide a more robust description of most of the dataset.

This connection between sparse optimization and "robust PCA" was made by Candès *et al.* [14], who also provided a tractable, convex approximation of the NP-hard problem (1):

$$\min_{L,S} \sum_i \sigma_i(L) + \lambda \|S\|_1, \text{ subject to } L + S = D. \quad (2)$$

The first term is the ℓ^1 norm of the vector $\sigma(L)$ of singular values of L, and is known as the *nuclear norm* or *Schatten 1-norm* of L. Further work by Lin *et al.* [15] developed a more efficient algorithm, using an alternating direction method of multipliers (ADMM) approach (see below).

Motivated by previous results in compressive sensing, in this work we will consider a nonconvex analog of (2), including an efficient ADMM algorithm. For the case when the data in D is substantially noisy, we will also incorporate additional (nonconvex) regularization, by penalizing oscillation in the sparse component S. This leads to a more formidable optimization problem to solve, for which we will combine an aggressive splitting method with ADMM.

In the next section, we will derive our nonconvex objective functions. We use these functions within an ADMM algorithm in Section III. In Section IV, we add additional regularization and describe the splitting approach. We test our algorithms on a synthetic video clip, and a real video clip with and without added noise in Section V.

Manuscript received August 16, 2011; revised January 17, 2012 and July 02, 2012; accepted July 07, 2012. Date of publication July 16, 2012; date of current version October 09, 2012. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jean-Christophe Pesquet. The author gratefully acknowledges the support of the U.S. Department of Energy through the LANL/LDRD Program for this work.

The author is with the Theoretical Division, MS B284, Los Alamos National Laboratory, Los Alamos, NM, 87545 USA (e-mail: rickc@lanl.gov).

This paper has supplemental downloadable multimedia material provided by the author, available at http://ieeexplore.ieee.org. This includes all video clip examples and results depicted in this paper. Each video clip is between 4 and 5 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2012.2208955

A portion of this work appeared in our conference proceedings [8], with fewer details and in the context of compressive sensing.

II. A New Nonconvex Objective Function and p-Shrinkage

A. Shrinkage

For the moment, consider the following simpler problem:

$$H_{\mu,1}(T) := \min_S \|S\|_1 + 1/(2\mu)\|S-T\|_F^2, \tag{3}$$

with $\mu > 0$. Notice that we obtain a problem of this form if in (2) we both fix L, and replace the equality constraint with a data fidelity term. Also, (3) is *separable*, meaning it can be solved at each matrix entry independently: $H_{\mu,1}(T) = \sum_{i,j} h_{\mu,1}(t_{ij})$, where

$$h_{\mu,1}(t) = \min_s |s| + 1/(2\mu)|s-t|^2. \tag{4}$$

(Later, we will consider the matrix ∇S containing a vector in each entry, so here and henceforth we allow the case that t and s are vectors.) This is known as the *Moreau envelope* of $|\cdot|$.

The (straightforward) solution of (4) is well known to be given by the following *shrinkage* operation (also known as *soft thresholding*):

$$s^* = \text{shrink}_1(t,\mu) := \max\{0, |t|-\mu\}\, t/|t|. \tag{5}$$

This is the *proximal function* of $|\cdot|$. The shrinkage operation is efficient, and when applied entrywise to matrices is trivially parallelizable.

The function of t defined by (4) is also straightforward to compute, and is a Huber function:

$$h_{\mu,1}(t) = \begin{cases} \frac{|t|^2}{2\mu}, & \text{if } |t| \le \mu, \\ |t| - \mu/2, & \text{if } |t| \ge \mu. \end{cases} \tag{6}$$

B. Nonconvex Generalization

Now we seek a generalization of (3) that is suitable for nonconvex optimization, while continuing to be efficiently solvable. In particular, we will construct a nonconvex function g such that the optimization problem

$$\min_s g(s) + 1/(2\mu)|s-t|^2 \tag{7}$$

can be solved by means of a generalization of the shrinkage operation (5).

It would seem that a natural way to generalize (3) to the nonconvex setting is to replace the ℓ^1 norm with the ℓ^p quasi-norm, with $0 < p < 1$. This approach is taken by Krishnan and Fergus in [16] in the context of image deblurring. However, the solution will no longer be given by a shrinkage. Moreover, the solution can not even be written explicitly, except for a few special values of p (such as $p = \frac{1}{2}$, which leads to a cubic equation).

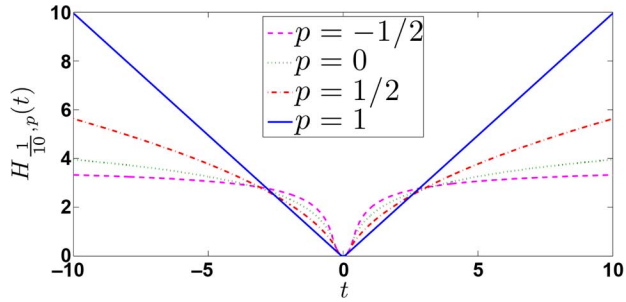

Fig. 1. Examples of the p-Huber function, smoothly splicing a quadratic with a pth-power function.

Instead, we take the approach of [8], and generalize the Huber function instead:

$$h_{\mu,p}(t) = \begin{cases} \frac{|t|^2}{2\mu}, & \text{if } |t| \le \mu^{\frac{1}{2-p}}, \\ |t|^p/p - \delta & \text{if } |t| \ge \mu^{\frac{1}{2-p}}, \end{cases} \tag{8}$$

where $\delta = \left(\frac{1}{p} - \frac{1}{2}\right)\mu^{\frac{p}{2-p}}$ is computed to make $H_{\mu,p}$ a C^1 function for all $p \in \mathbb{R}$. (When $p = 0$, we interpret (8) in a weak sense, and understand $\frac{|t|^p}{p}$ to mean $\log(|t|)$, with $\delta = \frac{\log\mu - 1}{2}$.) See Fig. 1 for examples. Note that when $p < 0$, $h_{\mu,p}(t)$ is bounded above.

Now we need to express $h_{\mu,p}$ as a Moreau envelope of a function $g_{\mu,p}$, as (4) does for the case of $p = 1$. We will do this using the Legendre-Fenchel transform, or convex conjugate, defined as follows: Given a function $f : \mathbb{R}^n \to \mathbb{R}$, its Legendre-Fenchel transform $f^* : \mathbb{R}^n \to \bar{\mathbb{R}}$ is the extended-real-valued function given by:

$$f^*(y) = \max_x x \cdot y - f(x). \tag{9}$$

Now we define a function $g_{\mu,p}$ by:

$$|s|^2/2 + \mu g_{\mu,p}(s) = \left(|\cdot|^2/2 - \mu h_{\mu,p}\right)^*(s). \tag{10}$$

Now by construction, for $p \le 1$, $f_{\mu,p}(t) := \frac{|t|^2}{2} - \mu h_{\mu,p}(t)$ is convex (being a continuous splicing of 0 near the origin, and a convex function minus a concave function away from the origin; see Fig. 13. It is actually convex for $1 < p \le 2$ as well, but we shall not use this). Being continuous also, it follows from [17, Thm. 11.1] that $f_{\mu,p}$ is its own *biconjugate*, so that

$$\begin{aligned} f_{\mu,p}(t) = f^{**}_{\mu,p}(t) &= \left(|\cdot|^2/2 + \mu g_{\mu,p}\right)^*(t) \\ &= \max_s s \cdot t - |s|^2/2 - \mu g_{\mu,p}(s). \end{aligned} \tag{11}$$

Rearranging the quadratic terms, we obtain our desired formulation:

$$h_{\mu,p}(t) = \min_s g_{\mu,p}(s) + 1/(2\mu)|s-t|^2. \tag{12}$$

Definition 1: Let $g_{\mu,p}$ be defined by (10). We call the penalty function

$$G_{\mu,p}(S) := \sum_{\text{entries } s \text{ of } S} g_{\mu,p}(s) \tag{13}$$

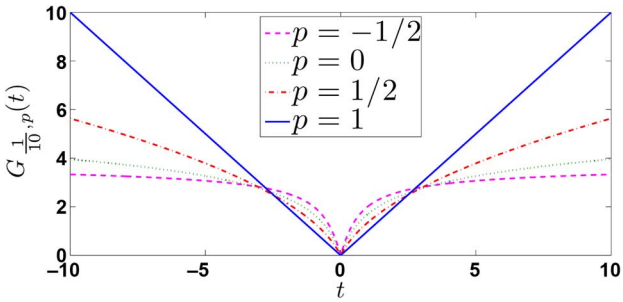

Fig. 2. Examples of $g_{\mu,p}$. It approaches $h_{\mu,p}(t)$ for large $|t|$, but has a sharp corner at $t = 0$.

the *proximal p-norm* of S, whether S is a vector, matrix, or multidimensional array.

This is a mild abuse of terminology, as $G_{\mu,p}$ possesses most but not all of the properties required of a norm (see Prop. 3). Now we give the most important property of $G_{\mu,p}$:

Proposition 2: The proximal function of $G_{\mu,p}$ is given entry-wise by the following *p-shrinkage* operation:

$$\text{shrink}_p(t,\mu) := \max\{0, |t| - \mu|t|^{p-1}\}\frac{t}{|t|}. \tag{14}$$

This proposition tells us that our proximal p-norm preserves the property of the ℓ^1 norm that optimization problems of the form (3) can be solved very efficiently. The proof is in the Appendix.

Now we record some properties of the proximal p-norm, which are established in the Appendix:

Proposition 3: Let $p \leq 1$. The penalty function $G_{\mu,p}$ is radial, radially strictly increasing, nonnegative, nonsmooth, continuous, and satisfies the triangle inequality.

The reader has likely by now surmised that we cannot in general write $g_{\mu,p}(t)$ explicitly. See Fig. 2 for numerically-computed examples. It is possible for a few values of p, such as $p = \frac{1}{2}$ where it can be expressed in terms of the solution of a cubic equation (much like the proximal function of the ℓ^p norm above). However, this will not matter, as being able to compute the proximal function of $g_{\mu,p}$ (that is, the solution of (12)) is all that will be necessary, and we can do so efficiently by means of p-shrinkage.

III. Nonconvex ADMM Algorithm

Now we can state our nonconvex approximation of (1):

$$\min_{L,S} G_{\mu,p}\left(\sigma(L)\right) + \lambda G_{\mu,p}(S), \text{ subject to } L + S = D, \tag{15}$$

where $\sigma(L)$ is the vector of singular values of L. As in [15], we use an ADMM algorithm to solve (15). First, we relax the equality constraint, and incorporate an additional variable Λ that can be interpreted as a Lagrange multiplier:

$$\min_{L,S} G_{\mu,p}\left(\sigma(L)\right) + \lambda G_{\mu\lambda,p}(S) + 1/(2\mu)\|D - L - S - \Lambda\|_F^2. \tag{16}$$

The *method of multipliers* will allow the equality constraint to hold at convergence. We proceed by alternating between fixing one of the variables and solving for the other.

With fixed L, solving for S is a simple p-shrinkage operation, thanks to the design of our objective function:

$$S^{n+1} = \text{shrink}_p(D - L^n - \Lambda^n, \mu\lambda). \tag{17}$$

Fixing S, we need to shrink $\sigma(L)$:

$$L^{n+1} = U\text{shrink}_p(\Sigma,\mu)V^T, \text{ where } U\Sigma V^T = D - S^{n+1} - \Lambda^n. \tag{18}$$

That is, the shrinkage operation can be passed inside the singular value decomposition (SVD), a fact established in the Appendix:

Proposition 4: Let the matrix A have SVD $A = U\Sigma V^T$. Then the solution of the optimization problem

$$\min_L G_{\mu,p}\left(\sigma(L)\right) + 1/(2\mu)\|L - A\|_F^2 \tag{19}$$

is given by $L = U\text{shrink}_p(\Sigma,\mu)V^T$.

(The case of $p = 1$ was proved in [18], but the proof in the Appendix is more straightforward as well as more general.)

Finally, we update Λ according to the method of multipliers:

$$\Lambda^{n+1} = \Lambda^n + S^{n+1} + L^{n+1} - D. \tag{20}$$

At each iteration, μ is decreased by a constant factor, subject to a lower bound.

In [15], the authors prove that when $p = 1$, the algorithm defined by (17), (18), and (20) converges. We cannot prove convergence in the nonconvex case, but empirically we have found the algorithm to converge reliably over a broad spectrum of examples. It can, on occasion, show mild oscillation when very near convergence, particularly for $p \ll 1$. We have found we can stabilize this by mollifying the $|t|^p$ term in (14), approximating $|t|$ with $\sqrt{t^2 + \epsilon}$. However, we have not found this to be necessary in any of the examples in this paper, and we defer detailed consideration of this mollification to future work.

IV. Regularization for Decomposition of Noisy Data

Now we consider the case of very noisy data. Clearly, we no longer wish to enforce the equality constraint. This requires only removing the Lagrange multiplier from (16). However, we find this insufficient, with too much noise remaining in the sparse component S.

To remedy this problem, we penalize oscillations in S, by adding a penalty function of ∇S, the discrete, 3-D gradient of S interpreted as a spacetime cube. We wish the penalty function to be nonsmooth, so that the frames of S can continue to have sharp edges. A nonconvex penalty function is also useful, as this is better able to preserve shapes and contrast [19]. We obtain the following problem:

$$\min_{L,S} G_{\mu,p}\left(\sigma(L)\right) + \lambda G_{\mu\lambda,p}(S) + \nu G_{\mu\nu,q}(\nabla S) + 1/(2\mu)\|D - L - S\|_F^2, \tag{21}$$

where $q \leq 1$.

The problem with (21) is that the problem of solving for S with fixed L is very difficult. We deal with this by applying a *splitting* approach, which can be seen as an extension of the approach of Wang *et al.* [20]. We replace both S and ∇S with

proxy variables X and Y, then add terms that relax the equality constraint with each proxy:

$$\begin{aligned}\min_{L,S,X,Y} & G_{\mu,p}(\sigma(L)) + \lambda G_{\alpha\lambda,p}(X) + 1/(2\alpha)\|X - S\|_F^2 \\ & + \nu G_{\beta\nu,q}(Y) + 1/(2\beta)\|Y - \nabla S\|_F^2 \\ & + 1/(2\mu)\|D - L - S\|_F^2. \end{aligned} \tag{22}$$

Furthermore, we wish the equality constraints $X = S$ and $Y = \nabla S$ to hold at convergence, so we introduce Λ_1 and Λ_2 for the method of multipliers:

$$\begin{aligned}\min_{L,S,X,Y} & G_{\mu,p}(\sigma(L)) + \lambda G_{\alpha\lambda,p}(X) + 1/(2\alpha)\|X - S - \Lambda_1\|_F^2 \\ & + \nu G_{\beta\nu,q}(Y) + 1/(2\beta)\|Y - \nabla S - \Lambda_2\|_F^2 \\ & + 1/(2\mu)\|D - L - S\|_F^2. \end{aligned} \tag{23}$$

The new variables X and Y can be solved for by a simple p- (or q-) shrinkage, and L can be solved for as before. The S subproblem is now quadratic, leading to the following linear equation:

$$\begin{aligned}\left((1/\alpha + 1/\mu)I + \frac{1}{\beta}\nabla^T\nabla\right) S = & \frac{1}{\alpha}(X - \Lambda_1) \\ & + \frac{1}{\mu}(D - L) + \frac{1}{\beta}\nabla^T(Y - \Lambda_2). \end{aligned} \tag{24}$$

As long as we use periodic boundary conditions in our discrete gradient ∇, the system matrix in (24) will be diagonalized by the discrete Fourier transform; that is, there is a diagonal matrix (or equivalently, an entrywise multiplication operator) K such that

$$\left(\frac{1}{\alpha} + \frac{1}{\mu}\right) I + \frac{1}{\beta}\nabla^T\nabla = \mathcal{F}_3^{-1} K \mathcal{F}_3. \tag{25}$$

Consequently, we can solve for S rapidly by simply taking the 3-D FFT of the right side of (24), dividing entrywise by the fixed kernel K, which can be precomputed (as it doesn't change during the iteration), and then taking the inverse FFT.

This gives us the iteration shown in (26) at the bottom of the page.

Fig. 3. Two frames of a synthetic video clip. The dark square moves from left to right until it reaches the position shown in the right frame, after which time it is stationary.

V. Numerical Experiments: Background Subtraction in Video

A. Noiseless Case

1) Synthetic Video: We apply the algorithm defined by (17), (18), and (20) to a dataset consisting of a video clip, with each frame comprising a column of our data matrix D. We begin with a synthetic clip, constructed to provide an example where the ground truth is known. We use a still image of an outdoor scene, and superimpose various shapes moving across the image, with different sizes and speeds, and with some shapes having fixed intensities while others have an intensity that is a fixed value greater or less than the background (see Fig. 3). We generate 256 frames, each of size 256 × 256, so our resulting matrix D has size 65536 × 256.

Since motion of objects within images is a nonlinear process, we expect a low-rank + sparse decomposition to place all of the moving objects into the sparse component S, so that what remains can have low rank. This is approximately what is observed in [14]. In our constructed video, a large square moves across the image, but then is stationary from frame 128 on. Thus, the optimal L^* should be of rank 2, consisting of the still image alone for the first 127 frames, followed by the still image with the superimposed square for the remaining frames. Then $D - L^* = S^*$ will consist of solely objects that are moving.

We compare the results of our algorithm with [15, Alg. 5], which corresponds to the $p = 1$ case above. Their code, written by M. Chen and A. Ganesh, initializes μ to be 4/5 times the

$$\begin{aligned} S^{n+1} &= \mathcal{F}_3^{-1}\left(\frac{\mathcal{F}_3\left[\frac{1}{\alpha}\left(X^n - \Lambda_1^n\right) + \frac{1}{\mu}(D - L^n) + \frac{1}{\beta}\nabla^T\left(Y^n - \Lambda_2^n\right)\right]}{K}\right); \\ L^{n+1} &= U\,\mathrm{shrink}_p(\Sigma, \mu)V^T, \text{ where } U\Sigma V^T = D - S^{n+1}; \\ X^{n+1} &= \mathrm{shrink}_p\left(S^{n+1} + \Lambda_1^n, \alpha\lambda\right); \\ Y^{n+1} &= \mathrm{shrink}_q\left(\nabla S^{n+1} + \Lambda_2^n, \beta\nu\right); \\ \Lambda_1^{n+1} &= \Lambda_1^n - X^{n+1} + S^{n+1}; \text{ and} \\ \Lambda_2^{n+1} &= \Lambda_2^n - Y^{n+1} + \nabla S^{n+1}. \end{aligned} \tag{26}$$

TABLE I
DECOMPOSITION RESULTS FOR VARYING p

p	λ	time (s)	rank(L)	$\|S\|_0$ (million)	SNR (dB)
1	2.4×10^{-3}	129	11	3.93	14.27
0.9	1.3×10^{-3}	207	6	3.01	15.99
0.8	7.4×10^{-4}	182	3	1.89	17.34
0.7	4.1×10^{-4}	181	2	1.65	18.28
0.6	2.4×10^{-4}	186	2	1.60	18.64
0.5	1.6×10^{-4}	184	2	1.62	18.15
0.4	1.1×10^{-4}	234	5	2.23	15.65
0.3	9.6×10^{-5}	267	14	3.22	10.75
0.2	8.6×10^{-5}	556	34	3.40	8.49
0.1	9.2×10^{-5}	341	91	2.27	6.17
0	8.9×10^{-5}	263	86	1.03	5.43

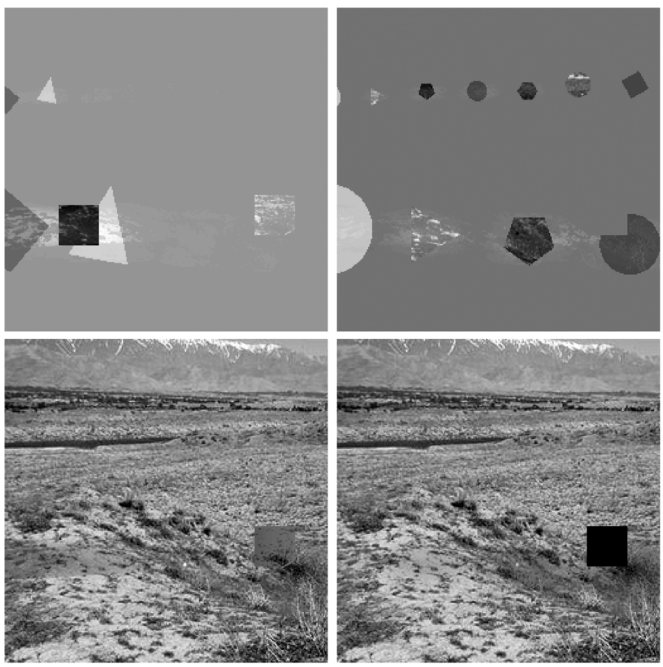

Fig. 4. Same two frames, from the sparse (top) and low rank (bottom) components, for the solution of (15) with $p = 1$. The sparse component contains traces of the background image, while the low rank component contains some of the dark square before it arrives in its final position.

largest singular value of D, giving $\mu_0 \approx 1630$. At each iteration μ is multiplied by 2/3, subject to a lower bound of $\mu_0 \times 10^{-7}$. The reduced SVD needed to compute L is computed in one of two ways, depending on the number r of singular values and vectors to be computed. If $r \geq 75$, Matlab's 'econ' SVD function is used, otherwise they are computed via Lanczos iterations using PROPACK (http://soi.stanford.edu/rmunk/PROPACK). The value of r is based on its previous value and the rank k of the previous iterate of L; if k was less than r, r is set to $k + 1$, whereas if r underestimated k (indicating that the rank grew quickly), r is set to $k + 14$. (The values 75 and 14 are based on heuristics depending on the number of columns of D.) The

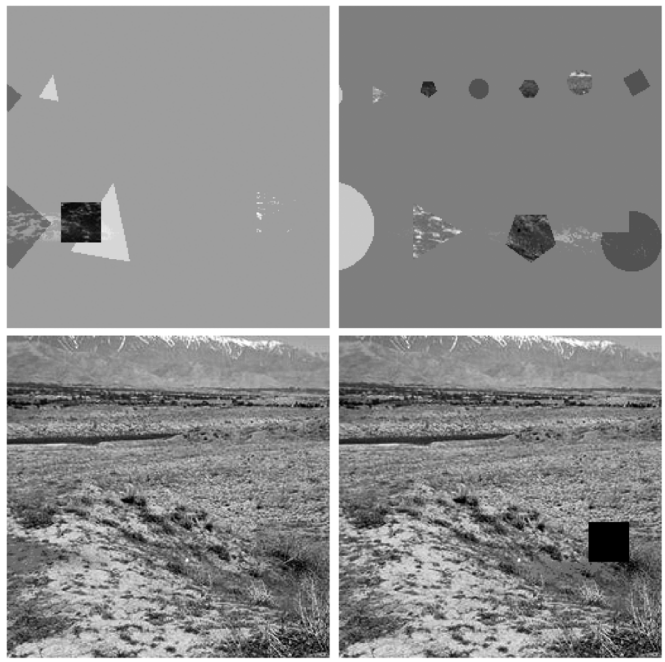

Fig. 5. As in Fig. 4, now with $p = 0.6$. The sparse component contains much less of the background image (except in "camouflage" shapes where it is expected), and the low-rank component is the best approximation to L^* among the values of p that were tested.

stopping condition is that $\frac{\|D-L-S\|_F}{\|D\|_F} < 10^{-6}$. For ease of comparison, we adopt these choices for our algorithm as well.

We consider the cases of $p \in \{1, 0.9, 0.8, \ldots, 0\}$. We measure the results by the SNR of the resulting L relative to the L^* described above. The parameter λ is chosen (by trial and error) in each case to maximize the SNR. The results are given in Table I, with a few cases shown in Figs. 4, 5, and 6. The videos corresponding to these and subsequent examples will be available for download in MPEG-4 format at http://ieeexplore.ieee.org.

We see that the SNR improves as p decreases from 1 to 0.6, then deteriorates quickly for $p < 0.5$. The same trend is observed in the rank of L, reaching an expected lowest value of 2 before increasing for small p. Similarly, the number of nonzero components is lower when the SNR is higher, except for very small p where $\|S\|_0$ decreases again.

The computation times are as computed on a two, 2.66 GHz core MacBook Pro with 8 GB of memory, running in Matlab (version R2010a). The time is shortest for $p = 1$, the one case where no exponentiation is required for the shrinkage. The computation time is also affected by the ranks of the iterates L^n and the number of iterations required. In Fig. 7 we plot the relative residual as a function of iteration for $p \in \{1, 0.6, 0.2\}$. The convergence is fastest for $p = 0.6$, somewhat slower for $p = 1$, and much slower for $p = 0.2$. In the $p = 0.2$ case the convergence begins to stagnate near our convergence threshold, though by this time the convergence is already adequate for practical applications.

2) Traffic Video: We also apply our approach to a real video clip, one obtained from a traffic camera. The clip was extracted from the Lankershim dataset (camera 4, 8:45–9:00 AM) available at http://ngsim-community.org. We consider a portion of

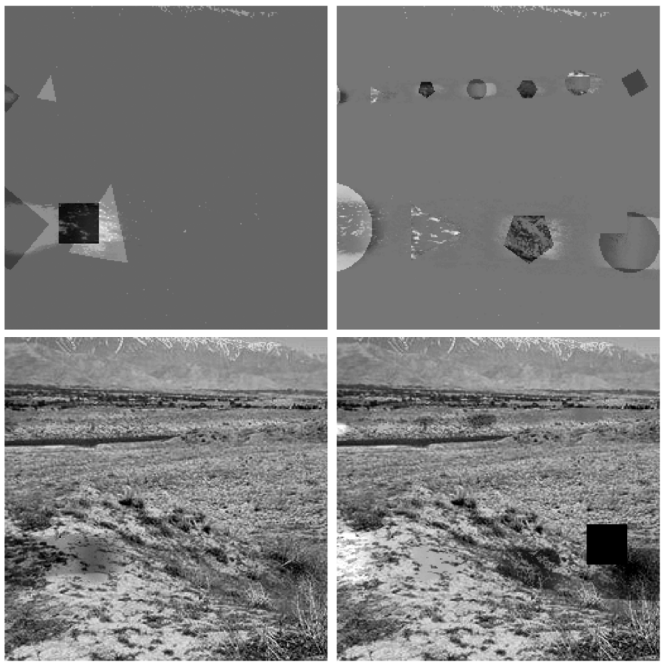

Fig. 6. As in Fig. 4, now with $p = 0.2$. The sparse component contains more of the background image, while the low rank component exhibits substantial "ghosting," faint objects moving across the image that should be in the sparse component instead.

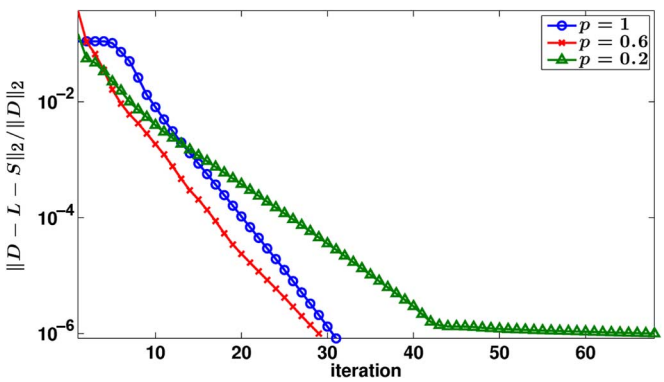

Fig. 7. Plot of the relative residual versus iteration, for three values of p. In addition to giving the most accurate reconstruction, $p = 0.6$ gives faster convergence than the convex case $p = 1$. Reducing p to 0.2 results in slower convergence, nearly stalling once the residual gets very small.

288 frames in length, with each frame converted to a grayscale image of size 240 × 320 pixels, taking values in [0, 1]. Our resulting data matrix D therefore is of size 76 800 × 288.

We compare the cases of $p = 1$ and $p = 0.6$, the latter having given the best result in the synthetic video. Without ground truth to compare with, we choose the parameter λ subjectively, to give the best separation between moving and stationary objects. In the $p = 1$ case, analysis in [14] suggests the value $\lambda = 76800^{\frac{-1}{2}}$ is appropriate, a choice corroborated by our tests. For $p = 0.6$, λ is manually tuned to give the best background subtraction, namely 5×10^{-4}. Selected frames are in Figs. 8 and 9.

In the $p = 1$ case, the algorithm converged in 33 iterations to an L of rank 145 and an S with 21 066 189 nonzero entries. The large number of nonzero entries is due to a small amount of noise in the video, all of which ends up S. There were many

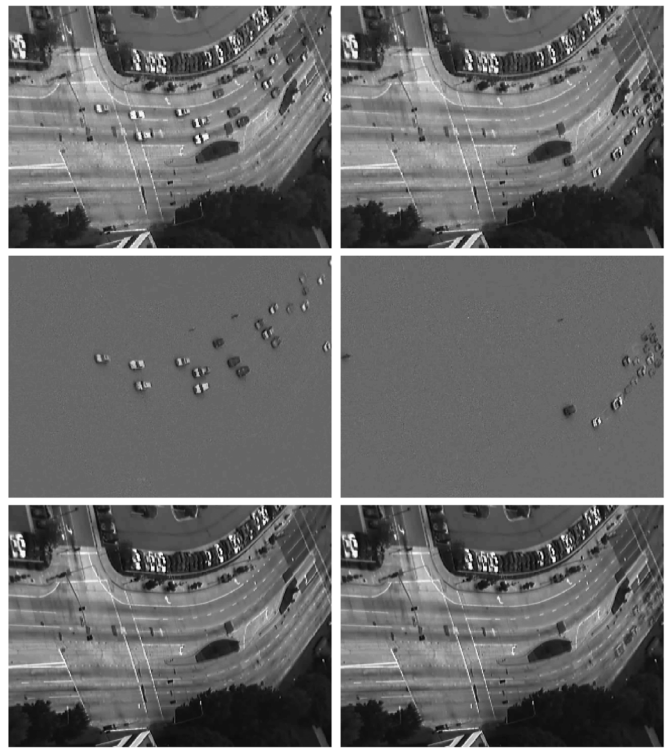

Fig. 8. Top: two frames from a traffic video clip. Middle: corresponding frame of S, the sparse component, for $p = 1$. S contains all moving objects, including a cyclist and pedestrian, except in the right frame some cars stopped at the stoplight appear as well. Bottom: corresponding frame of L, the low rank component. Primarily the stationary background, with occluded traffic lines filled in, but in the right frame the stationary cars are faint.

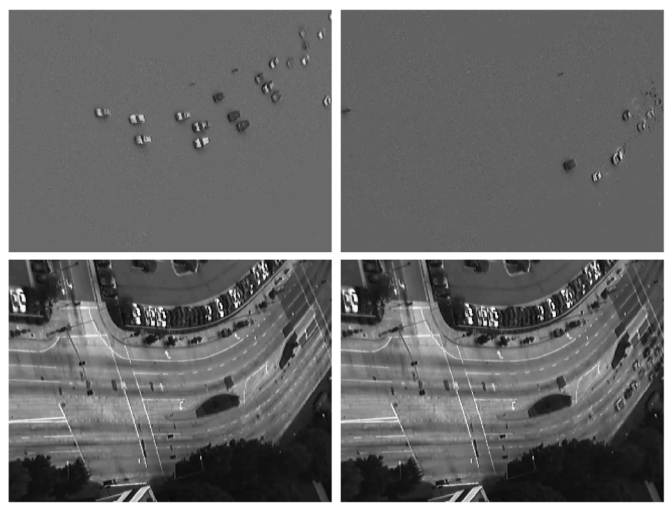

Fig. 9. Top: same two frames of S as in Fig. 8, except computed using $p = 0.6$. S contains all moving objects, with the only cars in the right frame being those still pulling up to the stoplight. Bottom: frames of L, the low rank component, containing the extracted background.

fewer entries of substantial size, with 439 719 entries having magnitude more than 0.1, or 2% of the total number of entries. The execution time in Matlab was 275 s on the laptop described above, or 202 s on an eight, 2.8 GHz core Linux machine with 64 GB memory. The sparse component S contains primarily the moving objects within the video, mostly cars but also a cyclist and a pedestrian. However, some cars that are stopped at a red

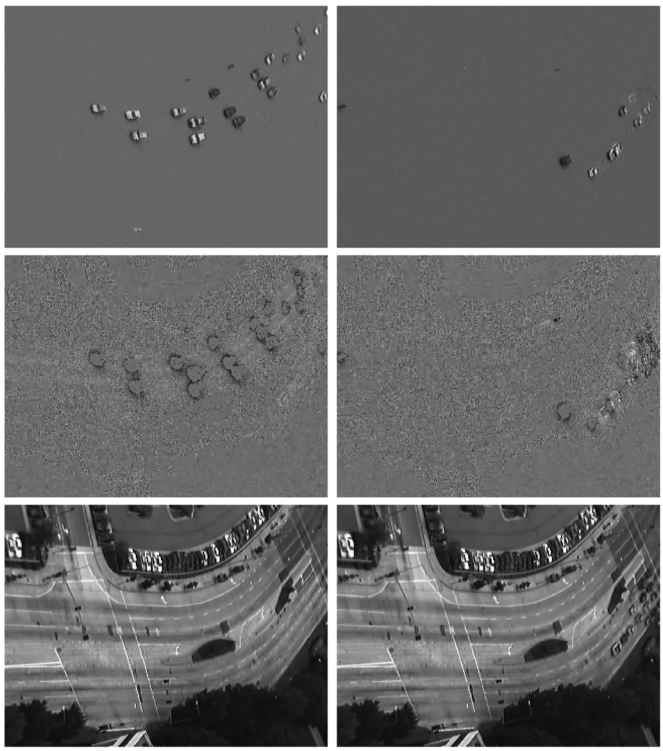

Fig. 10. Top: two frames of sparse component S, computed using (26) with $q = 1$. The noise is reduced, but not eliminated entirely. Middle: the residual $D - L - S$ contains mostly noise, but some signal leakage is present. Bottom: the low rank component L is entirely noisefree.

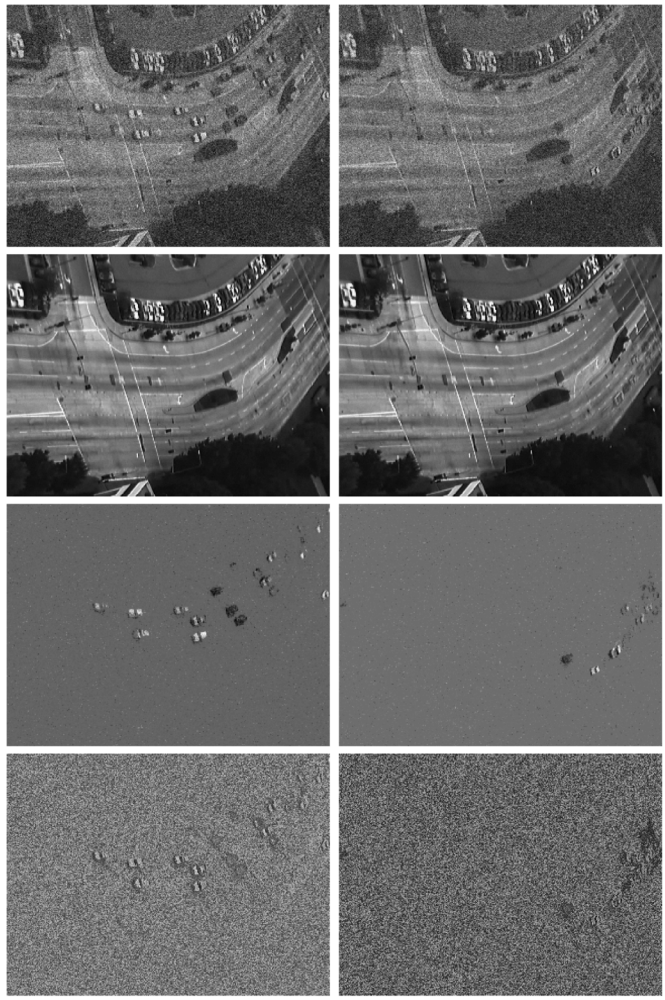

Fig. 11. Top: two frames from the noisy video. Second row: the background scene of the $q = 1$ reconstruction is essentially the same as in the previous experiment. Third row: the sparse component contains the moving objects as before, with less noise than in the original video. Bottom: the residual $D-L-S$. Some signal is present, at an intensity comparable to that of the noise.

light and then proceed upon green end up in S throughout. The low rank component L is mostly the stationary background, except for cars temporarily coming to a stop being absent.

In the $p = 0.6$ case, the algorithm converged in 39 iterations to an L of rank 12 and an S with 19 946 476 nonzero entries, with 437 923 of them having magnitude over 0.1 (or just under 2%). The execution time was 390 s on the MacBook Pro or 251 s on the eight-core machine. The background subtraction is generally similar to the $p = 1$ case, except that cars temporarily stopped are in L while stopped, and in S once they proceed. We thus obtain a cleaner separation between stationary and moving objects.

B. Example: Noisy Video

1) Actual Noise: We apply the algorithm (26) first to the video of the previous section, with the goal of removing the slight noise that is present. We use $p = 0.6$, and compare $q = 1$ and $q = 0.5$ (for the penalty function on ∇S). We find that for the low noise level that is present, the difference between the two is negligible, and we display only the $q = 1$ result in Fig. 10.

The parameters were chosen manually by trial and error, to provide a balance between a good separation between moving and stationary objects, reducing noise in S, while keeping the "signal" present in the residual $D - L - S$ to a level at most comparable to the magnitude of the noise. For both values of q, the values found were $\lambda = 0.3$ and $\mu = \alpha = \beta = 10$. The results are not very sensitive to α and β, whose values were left unchanged for all subsequent experiments. Changing q only requires rechoosing ν, which was chosen to be 8×10^{-7} for $q = 1$ and 1×10^{-7} for $q = 0.5$. Our previous stopping condition is no longer applicable, since the residual $\|D - L - S\|_F$ is no longer being driven to zero, so we just fix the number of iterations at a conservative 100.

As seen in Fig. 10, we are able to remove some of the noise, but a little noise remains. The residual $D - L - S$ shows the noise that was removed, as well as a little of the signal, particularly shadows of the cars. The low rank component L does not contain any visible noise (as with the equality constrained algorithm), as almost any noise would greatly increase the rank. The noise is stronger in the roads than elsewhere, suggesting Poisson noise (or signal-dependent Gaussian noise, given that photon counts are not very low). A Poisson log-likelihood term could give better results than the ℓ^2 norm for the residual, but we will not pursue this here.

As an alternative, we could have simply applied a denoising algorithm such as total-variation regularization. However, the current approach has the advantage of removing the background

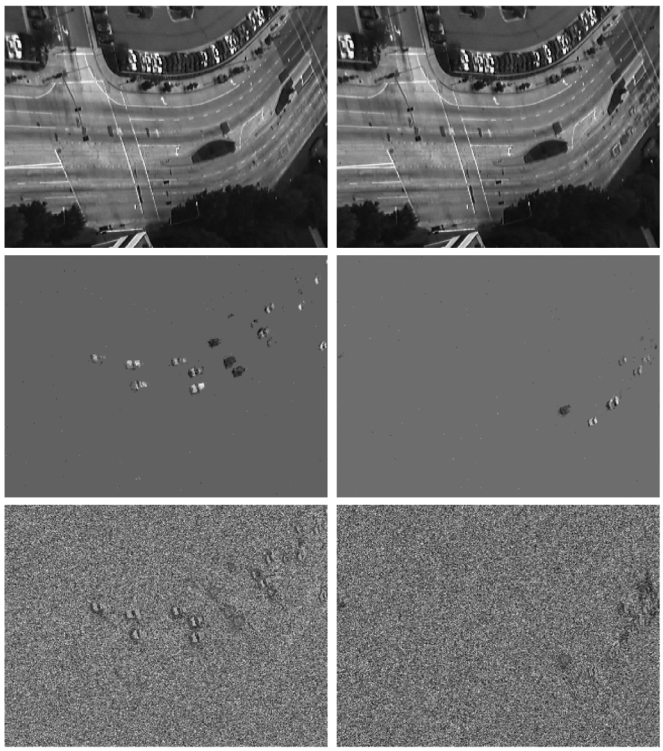

Fig. 12. Top: the background scene of the $q = \frac{-1}{2}$ reconstruction is essentially the same as in the previous experiment and the $q = 1$ case. Middle: the sparse component contains the moving objects as before, with less noise than the $q = 1$ case. Bottom: the residual $D - L - S$. Signal in the residual is similar to the $q = 1$ case, despite the better denoising.

from the regularization process, preventing the slight degradation of fine-scale features in L that would be inevitable with any denoising method.

2) Added Noise: Now we apply the algorithm (26) to the same traffic video, but now with Gaussian noise of standard deviation 0.1 added independently to every entry of D. We continue to use $p = 0.6$, and compare the cases of $q \in \{1, 0.75, 0.5, \ldots, -1\}$. The parameters are chosen as before to maintain good qualitative background subtraction, while regularizing S only to a degree that results in signal leakage into the residual $D - L - S$ to an extent commensurate with the noise level. Only ν needs to be rechosen for each q; the other parameters were $\lambda = 0.06$, $\mu = \frac{1}{6 \times 10^{-3}}$, and $\alpha = \beta = 10$. The additional variables make our simple Matlab implementation more memory intensive, requiring about 8 GB, so our tests were run on the eight-core machine described above.

In all cases, the resulting L had a rank of 2. The matrix S was completely dense, but this is because sparsity was directly enforced on the proxy variable X, and a residual difference remains. The fraction of entries of S having magnitude exceeding 0.1 ranged between 1% and 1.5%, with no clear pattern of dependence on q.

Table II reports the SNR of $L + S$ relative to that computed in Section V-B-1 from the original, low-noise video, this being the closest we have to a noisefree ground truth. The corresponding SNR of the original video is 29.15 dB, that of the noise-added video is 5.00 dB. We find that the best denoising performance is for $q = \frac{-1}{2}$, with an SNR of 17.98 dB, compared with 16.97 dB for $q = 1$. See Figs. 11 and 12. Recall that ν was chosen to give the same degree of signal leakage into the residual for all values of q; for smaller q, the better shape and contrast preservation allows the regularization to be stronger for the same degree of leakage.

TABLE II
REGULARIZED DECOMPOSITION RESULTS FOR VARYING q

q	ν	time (s)	SNR (dB)
1	7×10^{-5}	809	16.97
0.75	4×10^{-5}	861	17.55
0.5	2×10^{-5}	874	17.78
0.25	7×10^{-6}	866	17.81
0	2×10^{-6}	867	17.86
-0.25	5×10^{-7}	865	17.83
-0.5	2×10^{-7}	868	17.98
-0.75	3×10^{-8}	866	17.44
-1	9×10^{-9}	822	17.57

VI. CONCLUSION

We presented two nonconvex optimization problems for sparse matrix optimization, based on a novel nonconvex penalty that lends itself to efficient minimization via a generalized shrinkage. Our nonconvex generalization of the robust PCA approach of [15] produces a sparser model that is better able to distinguish between moving and stationary objects. Combined with a nonconvex regularization, our splitting algorithm can maintain the background subtraction while removing substantial noise, more so than convex regularization can.

The area of parameter selection is in need of improvement, particularly for the algorithm of (26) with its five parameters. Work is currently ongoing with the authors of [21], in a more general, nonlinear context, to determine how to select parameters *a priori* where possible, and otherwise to develop heuristic strategies that can narrow the search. This work will also address the issue of stopping conditions. Future work will also include a massively parallel, GPU implementation of the algorithms presented here, something facilitated by the straightforward parallelizability of our approach.

APPENDIX A
PROOF OF PROPOSITION 2

Proof: The proximal function of $G_{\mu,p}$ is the solution of the following optimization problem:

$$\min_S G_{\mu,p}(S) + 1/(2\mu)\|S - T\|_F^2. \tag{27}$$

As both terms are sums of functions depending on a single matrix entry, the optimization problem is separable. Thus, the value of each entry of the proximal function is the minimizer s^* of (12). This is the same as the maximizer within the definition (11) of $f^{**}_{\mu,p}(t)$. By a standard result of convex duality [17, Prop. 11.3], this is given by $s^* = \nabla f_{\mu,p}(t)$, from which (14) follows by a straightforward computation. ■

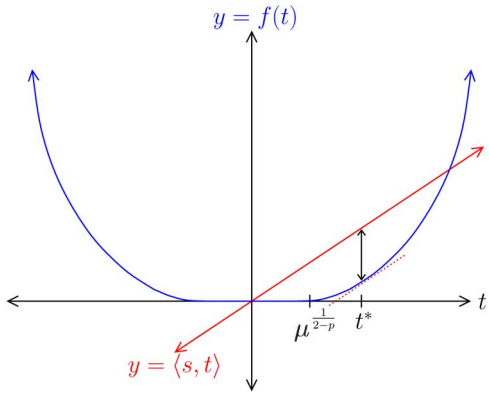

Fig. 13. An example of the function $f(t) = \frac{t^2}{2} - h_{\mu,p}(t)$. Its convex conjugate at s is the greatest height above the graph of f attained by a line through the origin with slope s.

Appendix B
Proof of Proposition 3

Proof: First, that $g_{\mu,p}$ is radial follows from (10) and the fact that $h_{\mu,p}$ is radial. Hence it suffices to consider the scalar function $g(s) = g_{\mu,p}(s)$ for $s \in \mathbb{R}$; we will suppress the dependence on μ and p for convenience.

The rest of the properties will follow from the basic properties of $f(t) := f_{\mu,p}(t)$ and the definition of the convex conjugate; see Fig. 13. A key relation, dual to one used above, is the following:

$$\arg\max_t \left(st - f(t)\right) = \partial\left(s^2/2 + \mu g(s)\right) = s + \mu \partial g(s). \quad (28)$$

The maximizer of $st - f(t)$ is unique except when $s = 0$, when $\arg\max -f(t) = [-\mu^{\frac{1}{2-p}}, \mu^{\frac{1}{2-p}}]$. Thus we have that g is differentiable except at 0 where it has a subdifferential (the set of subgradients, which are slopes of supporting "tangent" lines) of $[-\mu^{\frac{p-1}{2-p}}, \mu^{\frac{p-1}{2-p}}]$. Thus g is nonsmooth. This also tells us that g is continuous away from zero, but it is continuous at zero as well, since $g(0) = 0$, and as $s \to 0$, the maximal height of $s \cdot t$ above $f(t)$ approaches zero as well (see Fig. 13).

To show that $g(s)$ is increasing on $(0, \infty)$, we show that $g'(s) > 0$ there. By (28), we need that $\arg\max_t(st - f(t))$ exceeds s. Differentiating gives us the equation $t - \mu t^{p-1} = s$, the second term of which making it clear that the solution satisfies $t^* > s$. Thus $g_{\mu,p}(s)$ is an increasing function of $|s|$. Since $g(0) = 0$, this also gives us nonnegativity.

For the triangle inequality, we need to show that $g(x + y) \leq g(x) + g(y)$. If $xy = 0$, this is trivial. If $xy < 0$, then $|x + y| < \max\{|x|, |y|\}$. Hence

$$g(x + y) = g\left(|x + y|\right) < g\left(\max\left\{|x|, |y|\right\}\right) < g(x) + g(y). \quad (29)$$

Lastly, assume $xy > 0$, and without loss of generality that x and y are positive. If we can show that g' is decreasing on $(0, \infty)$ (i.e., that g is concave there), the triangle inequality will follow immediately from $\int_0^y g'(x + t) - g'(t)dt \leq 0$ and the fundamental theorem. We have from (28) that $s + \mu g'(s) = t^*$, where t^* satisfies $t^* - \mu(t^*)^{p-1} = s$. Thus $\mu g'(s) = t^* - s = \mu(t^*)^{p-1}$. It is clear (see Fig. 13) that for $s > 0$, t^* is a positive, increasing function of s. Hence $(t^*)^{p-1}$ is a decreasing function of s, thus $g'(s)$ is also, completing the proof. ∎

Appendix C
Proof of Proposition 4

Proof: The Frobenius norm is unitarily invariant, and the expression $G_{\mu,p}(\sigma(L))$ is as well, depending on only the singular values of L. We thus have that the following problem is equivalent to (19):

$$\min_L G_{\mu,p}\left(\sigma(L)\right) + \frac{1}{2\mu}\|U^T L V - \Sigma\|_F^2. \quad (30)$$

If we consider the SVD of L as well, say $L = W\Theta Z^T$, we can reformulate (30):

$$\min_{\text{diagonal } \Theta, \text{unitary } W,Z} G_{\mu,p}(\Theta) + \frac{1}{2\mu}\|U^T W \Theta Z^T V - \Sigma\|_F^2, \quad (31)$$

noting that $G_{\mu,p}(\sigma(W\Theta Z^T)) = G_{\mu,p}(\Theta)$, and where "diagonal" is in the usual sense in the context of the SVD for rectangular matrices. Now, it is shown in Example 7.4.13 of [22] (see particularly (7.4.15)) that the Frobenius-norm term is minimized when the unitary factors $U^T W$ and $Z^T V$ are identity matrices. Consequently, the problem reduces to one involving diagonal matrices alone,

$$\min_{\text{diagonal } \Theta} G_{\mu,p}(\Theta) + \frac{1}{2\mu}\|\Sigma - \Theta\|_F^2. \quad (32)$$

The solution is $\Theta = \text{shrink}_p(\Sigma, \mu)$ by Prop. 2, giving $L = U\text{shrink}_p(\Sigma, \mu)V^T$. ∎

Acknowledgment

The author would like to thank Brendt Wohlberg and James Theiler for helpful discussions, and the anonymous reviewers for their suggestions that substantially improved this paper.

References

[1] Compressive sensing resources, Main CS repository [Online]. Available: http://www.dsp.ece.rice.edu/cs
[2] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, 2006.
[3] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
[4] R. Chartrand, "Exact reconstructions of sparse signals via nonconvex minimization," *IEEE Signal Process. Lett.*, vol. 14, no. 10, pp. 707–710, 2007.
[5] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *Proc. 33rd IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2008, pp. 3869–3872.
[6] R. Chartrand and V. Staneva, "Restricted isometry properties and nonconvex compressive sensing," *Inverse Problems*, vol. 24, no. 035020, pp. 1–14, 2008.
[7] R. Saab, R. Chartrand, and Ö Yilmaz, "Stable sparse approximations via nonconvex optimization," in *Proc. 33rd Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2008, pp. 3885–3888.
[8] R. Chartrand, "Fast algorithms for nonconvex compressive sensing: MRI reconstruction from very few data," in *Proc. IEEE Int. Symp. Biomed. Imag.*, 2009, pp. 262–265.
[9] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, pp. 717–772, 2009.
[10] M. Fazel, H. Hindi, and S. P. Boyd, "A rank minimization heuristic with application to minimum order system approximation," in *Proc. Amer. Control Conf.*, 2001, vol. 6, pp. 4734–4739.

[11] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 2080–2088.

[12] K. Min, Z. Zhang, J. Wright, and Y. Ma, "Decomposing background topics from keywords by principal component pursuit," in *Proc. ACM Int. Conf. Inf. Knowl. Manag.*, 2010, pp. 269–278.

[13] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images," in *Proc. IEEE Comput. Vis. Pattern Recogn.*, 2010, pp. 763–770.

[14] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *J. ACM*, vol. 58, 2011, article 11, 37 pp.

[15] Z. Lin, M. Chen, and Y. Ma, "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices," Univ. of Illinois-Urbana, Champaign, IL, Tech. Rep. UILU-ENG-09-2214, 2010.

[16] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-Laplacian priors," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1033–1041.

[17] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*. Berlin, Germany: Springer-Verlag, 1998.

[18] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM J. Optim.*, vol. 20, pp. 1956–1982, 2010.

[19] R. Chartrand, "Nonconvex regularization for shape preservation," in *Proc. IEEE Int. Conf. Image Process.*, 2007, pp. I-293–I-296.

[20] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, pp. 248–272, 2008.

[21] B. Wohlberg, R. Chartrand, and J. Theiler, "Local principal component pursuit for nonlinear datasets," in *Proc. 33rd IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2012, pp. 3925–3928.

[22] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

Rick Chartrand (M'06–SM'12) received the B.Sc. (Hons.) degree in mathematics from the University of Manitoba, Winnipeg, MB, Canada, in 1993 and the Ph.D. degree in mathematics from the University of California, Berkeley, in 1999.

He has held academic positions at Middlebury College and the University of Illinois at Chicago, before joining the Los Alamos National Laboratory, Los Alamos, NM, in 2003, where he is currently a Technical Staff Member in the Applied Mathematics and Plasma Physics group. His research interests include compressive sensing, nonconvex continuous optimization, image processing, dictionary learning, computing on accelerated platforms, and geometric modeling of high-dimensional data.